ภาษาที่ใช้และการพัฒนาโปรแกรมคอมพิวเตอร์

บทที่ 6

ภาษาคอมพิวเตอร์

ภาษาของคอมพิวเตอร์นั้น มีหลายภาษา ซึ่งแต่ละภาษาจะมีเอกลักษณ์เป็นของตัวเอง แตกต่างกันไปตามการใช้งาน

ระดับของภาษา 5 ระดับ

1. ภาษาเครื่อง(Machine Language)
2. ภาษาสัญลักษณ์(Symbolic Language) หรือ ภาษาแอสเซมบลี(Assembly Language)
3. ภาษาระดับสูง(High Level Language)

4. ภาษาระดับที่ 4 (4th GL : Fourth General Language)
5. ภาษาธรรมชาติ(Natural Language)

ภาษาเครื่อง(Machine Language)

- เป็นภาษาระดับต่ำ(Low Level Language)
- เป็นภาษาคอมพิวเตอร์ภาษาเดียวที่เครื่องคอมพิวเตอร์สามารถเข้าใจโดยไม่ต้องมีการแปล
- มีรูปแบบเป็นรหัสคำสั่งเป็นตัวเลขประกอบด้วย 0 และ 1 เช่น 1010 0001 เป็นต้น
- เป็นภาษาใช้งานเฉพาะเจาะจงกับเครื่องคอมพิวเตอร์หรือหน่วยประมวลผลกลาง ขึ้นอยู่กับสถาปัตยกรรมของเครื่องใดเครื่องหนึ่ง

ภาษาสัญลักษณ์(Symbolic Language) หรือ ภาษาแอสเซมบลี(Assembly Language)

- มีรูปแบบเป็นสัญลักษณ์แทนรหัสคำสั่งในภาษาเครื่อง ทำให้เข้าใจความหมายของคำสั่ง เช่น
  - ADD แทนการบวกเลข
  - SUB แทนการลบเลข
  - MUL แทนการคูณเลข
  - DIV แทนการหารเลข
  - MOV แทนการย้ายข้อมูล
- เป็นภาษาที่ขึ้นอยู่กับเครื่องคอมพิวเตอร์หรือหน่วยประมวลผลกลาง ขึ้นอยู่กับสถาปัตยกรรมของเครื่องใดเครื่องหนึ่ง เหมือนกับภาษาเครื่อง
- มีตัวแปลภาษาแอสเซมบลีไปเป็นภาษาเครื่อง เรียกว่า Assembler

ภาษาระดับสูง(High Level Language)

- เป็นภาษาไม่ขึ้นอยู่กับเครื่องคอมพิวเตอร์หรือหน่วยประมวลผลกลาง ยี่ห้อใดยี่ห้อหนึ่ง
- มีการเขียนโปรแกรมคล้ายคลึงกับการเขียนประโยคในภาษาอังกฤษ
- มีตัวแปลภาษาระดับสูงไปเป็นภาษาเครื่อง 2 ประเภทคือ Interpreter และ Compiler
- เหมาะสำหรับในการพัฒนาโปรแกรมใช้งาน และประยุกต์ใช้งานด้านต่าง ๆ ได้

กลุ่มภาษาระดับสูง (1/2)

- กลุ่มใช้งานด้านวิทยาศาสตร์ วิศวกรรมศาสตร์ งานวิจัย (Science) ได้แก่ ภาษาฟอร์แทรน(FORTRAN) และภาษาแอลกอล(ALGOL)
- กลุ่มใช้งานด้านธุรกิจ (Business) ได้แก่ ภาษาโคบอล (COBOL) ภาษาพีแอลวัน(PL/I) ภาษาอาร์พีจี(RPG)

กลุ่มภาษาระดับสูง (2/2)

- กลุ่มใช้งานทั่วไป(General Purpose) ได้แก่ ภาษาเบสิค(BASIC) ภาษาปาสคาล(Pascal) ภาษาซี(C Language)
- กลุ่มใช้งานด้านตรรกะ ได้แก่ ภาษาเอด้า(Ada) ภาษาโปรล็อก(Prolog) ภาษาสโนบอล(SNOBOL)

ภาษาฟอร์แทรน(FORmula TRANslator)

- เป็นภาษาระดับสูงภาษาแรก
- เหมาะสำหรับใช้งานด้านวิทยาศาสตร์ วิศวกรรมศาสตร์

ภาษาเบสิค(Beginner's All-purpose Symbolic Instruction Code)

- เป็นภาษาที่ใช้กันมากที่สุดสำหรับไมโครคอมพิวเตอร์ มินิคอมพิวเตอร์ และเมนเฟรมคอมพิวเตอร์
- ผู้คิดค้นคือ Dartmouth College
- เป็นภาษาที่ใช้โต้ตอบไปมาระหว่างกันได้ทันที จึงง่ายสำหรับผู้ที่ใช้ที่เริ่มต้นเรียน หรือต้องการจะเรียนรู้หรือทำการแก้ไขและแก้ไขซินแทกซ์เออเร่อส์ได้ง่าย

ภาษาโคบอล(COmmon Business Oriented Language)

- เหมาะที่จะใช้กับการประมวลผลข้อมูลทางธุรกิจ และการสร้างแฟ้มข้อมูลสำหรับงานต่างๆ
- ใช้ได้กับระบบคอมพิวเตอร์ทั่วๆ ไป

ภาษาปาสคาล(Pascal)

- ตั้งชื่อเป็นเกียรติแก่ Blaise Pascal
- ผู้คิดค้นคือ Nicklaus Wirth
- เป็นการเขียนโปรแกรมภาษาโครงสร้าง(Structure Programming) เพื่อให้โปรแกรมมีความเป็นมาตรฐานและง่ายแต่การแก้ไขโดยใช้คำสั่ง IF-THEN-ELSE และ DO-WHILE
- เป็นภาษาที่ใช้กับเครื่องไมโครคอมพิวเตอร์

ภาษาพีแอลวัน (PL/I : Programming Language/1)

- เป็นภาษาที่รวมเอาข้อดีของภาษาฟอร์แทนเข้ามารวมกับข้อดีของภาษาโคบอล

ภาษาซี(C Language)

- ผู้คิดค้นได้แก่ BELL LABS
- ภาษาซีจะใช้ในการจัดทำโอเอสและโปรแกรมระบบงานสำหรับงานด้านวิจัยและธุรกิจ

ภาษาฟอร์ท(Fourth)

- ใช้กับงานด้านวิศวกรรม
- เริ่มแรกเรียกว่า Fourth เพราะว่าเป็น Fourth-Generation Language และใช้ IBM1130 ในการประมวลผล ซึ่งยินยอมให้ใช้เพียง 5 ตัวอักษร สำหรับการตั้งชื่อใดๆ จึงต้องเปลี่ยนมาเป็น forth โดยปริยาย
- เหมาะสำหรับระบบเล็กๆ สามารถจัดทำโดยโอเอสได้และมักจะใช้กับคอมพิวเตอร์ขนาดเล็ก

ภาษาโลโก้(Logo)

- ใช้สอนเด็กในการฝึกหัดโต้ตอบกับเครื่องคอมพิวเตอร์
- โลโก้ใช้ในการจัดทำกราฟฟิก วาดภาพได้
- ใช้กันมากในโรงเรียน
- ใช้กับไมโครคอมพิวเตอร์

ภาษาซิมูเลชั่น

- ภาษาซิมมูเลชั่นที่ใช้ในการจัดทำโมเดล จะได้แก่ GPSS ที่มีชื่อเต็มว่า General Purpose System Simulator และ SIMULA ซึ่งมีชื่อย่อมาจาก Simulation Language

ภาษาระดับที่ 4 (4th GL : Fourth General Language)

เป็นการพัฒนาระบบโปรแกรมหรืออุตสาหกรรมซอฟต์แวร์ ที่ผู้พัฒนา(Developer) ใช้เครื่องมือสร้างโปรแกรม(Tools) โดยการออกแบบ(Design) ตามความต้องการแล้วสร้าง(Generate)ออกมาเป็นระบบโปรแกรมใช้งานได้เลย ทำให้สามารถพัฒนาโปรแกรมได้มีการทำงานซับซ้อนได้ สะดวกรวดเร็ว แก้ไขตรวจสอบได้

ภาษาธรรมชาติ(Natural Language)

- เป็นภาษาระดับที่ 5 ที่มนุษย์ได้พัฒนาฮาร์ดแวร์และซอฟต์แวร์ให้เครื่องคอมพิวเตอร์มีความสามารถเข้าใจภาษามนุษย์
- เป็นการทำให้คอมพิวเตอร์มีความเฉลียวฉลาดที่เป็นปัญญาประดิษฐ์(AI : Artificial Intelligence) เช่น แปลภาษาต่าง ๆ ของมนุษย์ได้ เช่น อังกฤษ ฝรั่งเศส ญี่ปุ่น เป็นต้น

การพัฒนาโปรแกรมคอมพิวเตอร์

การพัฒนาโปรแกรมคอมพิวเตอร์

- **ขั้นตอนที่เหมือนกัน เรียกว่า วงจรการพัฒนาระบบ (System Development Life Cycle :SDLC)**
- **ขั้นตอนการพัฒนาระบบมีอยู่ด้วยกัน 7 ขั้น**
  - เข้าใจปัญหา (Problem Recognition)
  - ศึกษาความเป็นไปได้ (Feasibility Study)
  - วิเคราะห์ (Analysis)
  - ออกแบบ (Design)
  - สร้างหรือพัฒนาระบบ (Construction)
  - การปรับเปลี่ยน (Conversion)
  - บำรุงรักษา (Maintenance)

เข้าใจปัญหา Problem Recognition

ผู้บริหารหรือผู้ใช้ต้องเข้าใจและทราบปัญหาของระบบงานเดิม จึงต้องการระบบสารสนเทศใหม่เพื่อแก้ไขระบบงานเดิม รวมไปถึงเพื่อเพิ่มประสิทธิภาพในการทำงาน, ให้การทำงานรวดเร็วมากขึ้น

ศึกษาความเป็นไปได้ Feasibility Study

- ต้องกำหนดให้ได้ว่าปัญหาคืออะไร
- ตัดสินใจสร้างระบบสารสนเทศใหม่ หรือแก้ไขระบบสารสนเทศเดิม
- ต้องกำหนดว่าการแก้ไขปัญหาเป็นทางเทคนิคหรือบุคลากร
- ความเป็นไปได้เรื่องค่าใช้จ่าย, ระยะเวลาในการพัฒนาระบบ และที่สำคัญคือ ผลประโยชน์ที่จะได้รับจากการลงทุน

วิเคราะห์ Analysis (1/3)

- การศึกษาระบบการทำงานของธุรกิจนั้นๆ
- กำหนดความต้องการของระบบใหม่ ใช้เทคนิคในการเก็บข้อมูล (Fact-Gathering Techniques)
- นำข้อมูลที่รวบรวมเขียนรายงานการทำงานของระบบเป็นตัวหนังสือหรือแสดงแผนภาพ

วิเคราะห์ Analysis (2/3)

- **เครื่องมือแสดงรายงาน**
  - ภาษาที่พัฒนาแบบ Structural language ได้แก่
    - พจนานุกรมข้อมูล (Data Dictionary)
    - แผนภาพการไหลของข้อมูล (Data Flow Diagram)
    - ข้อมูลเฉพาะการประมวลผล (Process Specification)
    - รูปแบบข้อมูล (Data Model)
    - รูปแบบระบบ (System Model)

วิเคราะห์ Analysis (3/3)

- **เครื่องมือแสดงรายงาน**
  - ภาษาการโปรแกรมเชิงวัตถุ(Object Oriented Programming) ได้แก่ Unified Modeling Language(UML)

ผังงาน(Flowchart) เป็นเครื่องมือที่ใช้อธิบายการทำงานของโปรแกรมได้ทั้งโปรแกรมที่พัฒนาด้วยภาษา Structural language และภาษาการโปรแกรมเชิงวัตถุ

ออกแบบ Design

- ขั้นตอนการวิเคราะห์ นักวิเคราะห์ระบบต้องหาว่า "จะต้องทำอะไร (What)" แต่ในขั้นตอนการออกแบบต้องรู้ว่า " จะต้องทำอย่างไร(How)"
- ได้แก่ ออกแบบฟอร์มสำหรับนำข้อมูลเข้า (Input Format), ออกแบบรายงาน (Report Format) และการแสดงผลบนจอภาพ (Screen Format)

สร้างหรือพัฒนาระบบ Construction

- เป็นขั้นตอนของโปรแกรมเมอร์
- เขียนโปรแกรมตามข้อมูลที่ได้จากเอกสาร
- ทดสอบโปรแกรมว่า ทำงานถูกต้องหรือไม่
- เตรียมคู่มือการใช้และฝึกอบรมผู้ที่จะใช้งานระบบจริง

การปรับเปลี่ยน Conversion

- การนำระบบงานใหม่มาใช้แทนระบบงานเก่าควรทำอย่างค่อยเป็นค่อยไปทีละน้อย
- วิธีที่ดีที่สุดคือ ใช้ระบบใหม่ควบคู่ไปกับระบบเก่าไปสักระยะหนึ่ง

บำรุงรักษา Maintenance

- การแก้ไขโปรแกรมหลังจากการใช้งานแล้ว
- สาเหตุที่ต้องแก้ไขมี 2 ข้อ คือ มีปัญหาในโปรแกรม (Bug) และ การดำเนินงานในองค์กรหรือธุรกิจเปลี่ยนไป

Algorithm

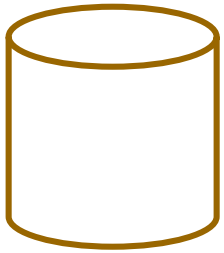
- **คือการแบ่งขั้นตอนการทำงานของโปรแกรมออกเป็นขั้นตอนย่อย ๆ ของการทำงานเพื่อให้เข้าใจง่ายขึ้น**
- **เช่น Algorithm ของการคำนวณภาษี 7 %**
  - ขั้นตอนที่ 1 รับจำนวนเงินที่ต้องการคำนวณ
  - ขั้นตอนที่ 2 กำหนดสูตรในการคำนวณภาษี

    (ภาษี = จำนวนเงิน * 0.07)
  - ขั้นตอนที่ 3 คำนวณจากสูตรที่กำหนด
  - ขั้นตอนที่ 4 แสดงผลลัพธ์ของการคำนวณ

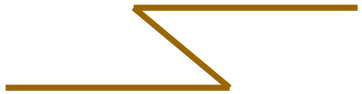
Flow Chart

- เป็นเครื่องมือที่โปรแกรมเมอร์ใช้ในการแปลง Algorithm หรือ ขั้นตอนการทำงานให้เป็นไปในลักษณะของแผนภาพเพื่อให้เข้าใจง่าย

Flow Chart Symbol



จานแม่เหล็ก (Magnetic disk) แสดงถึงการอินพุท / เอาท์พุทด้วยจานแม่เหล็ก

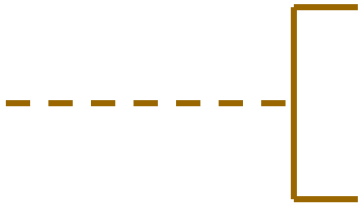


การติดต่อทางสาย (Communication link) แสดงถึงการส่งข่าวสารไปตามสายสื่อสาร

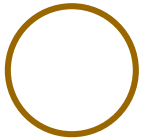
แสดงเส้นทางการไหล (Flow line) แสดงการเชื่อมต่อระหว่างสัญลักษณ์ 2 สัญลักษณ์

Flow Chart Symbol

การทำงานแบบขนาน (Parallel Mode)
แสดงการเริ่มหรือสิ้นสุดของการทำงาน
ตั้งแต่สองชนิดที่เกิดขึ้นพร้อมกัน



หมายเหตุ (Comment) แสดงถึง คำอธิบาย
หมายเหตุต่าง ๆ เพื่อให้เข้าใจยิ่งขึ้น

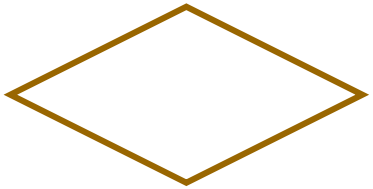


คอนเนคเตอร์ (Connector) แสดงการ
เชื่อมต่อของโฟลว์ชาร์ต ออกไปยังจุดอื่น ๆ

Flow Chart Symbol

จุดปลาย (Terminal) แสดงการเริ่มหรือสิ้นสุดของการทำงานของโปรแกรม

การประมวลผล (Process) ใช้กับการประมวลผลทุกชนิด



การตัดสินใจ (Decision) แสดงถึงการตัดสินเงื่อนไขว่าเป็นจริงหรือเท็จ

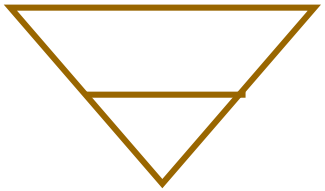
Flow Chart Symbol

การเตรียม (Preparation) ใช้ในการกำหนดค่าเริ่มต้นของการทำงาน

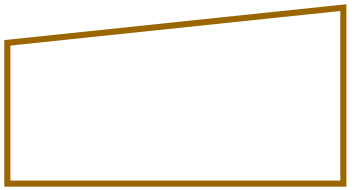
การประมวลผลที่กำหนดไว้ก่อน (Predefine Process) ใช้แทนการทำงานที่กำหนดไว้แล้ว ณ จุดใดจุดหนึ่งในโปรแกรม เช่นโปรแกรมย่อย

การควบคุมด้วยมือ (Manual Operation) เป็นการแสดงถึงการควบคุมการทำงานด้วยมนุษย์เข้าช่วยบางครั้ง

Flow Chart Symbol



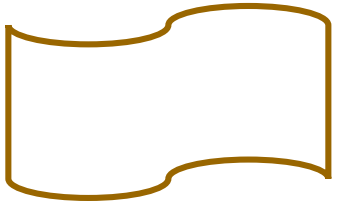
การใช้หน่วยความจำแบบออฟไลน์ (Off line storage) แสดงการเก็บข่าวสารในอุปกรณ์ความจำแบบออฟไลน์ใด ๆ



การป้อนข้อมูลด้วยมือ (Manual Input) เป็นการควบคุมการป้อนข้อมูลโดยผู้ใช้

เอกสาร (Document) แสดงถึงการอินพุท / เอาท์พุท ข้อมูลประเภทเอกสาร

Flow Chart Symbol

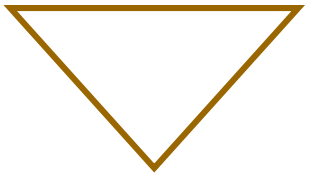


เทปกระดาษ (Punched Tape) แสดงถึงการอินพุท / เอาท์พุท ข้อมูลด้วยเทปกระดาษ

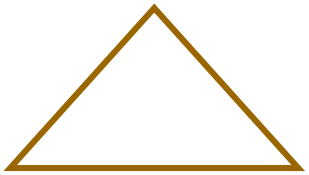
อินพุท / เอาท์พุท (Input / Output) แสดงถึงการป้อนข่าวสารที่ประมวลผลได้หรือแสดงผลที่เกิดจากการประมวลผล

เทปแม่เหล็ก (Magnetic Tape) แสดงถึงการอินพุท / เอาท์พุท ข้อมูลด้วยเทปแม่เหล็ก

Flow Chart Symbol



การรวม (Merge) แสดงถึงการนำเอาเซตของการบันทึกรวมกันเป็นชุดเดียว



การแยก (Extract) แสดงการแยกเซตของการบันทึกออกเป็นชุดที่ต้องการ

Flow Chart Symbol

การเก็บแบบออนไลน์ (Online storage) การใช้อุปกรณ์ความจำแบบออนไลน์ในการอินพุท / เอาท์พุท เช่น เทปแม่เหล็ก หรือ จานแม่เหล็ก

การแสดงผล (Display) การแสดงข่าวสารขณะประมวลผลผ่านทางอุปกรณ์ต่าง ๆ เช่น จอภาพ

เชื่อมต่อหน้ากระดาษ (Page Connector) แสดงการต่อของผังงานที่อยู่คนละหน้า

Flow Chart guidelines(1/2)

- การทำงานจะเริ่มจากบนลงล่าง และ จากซ้ายไปขวาของภาพ
- กิจกรรมที่เกิดใน Flow chart จะถูกกำหนดไว้อย่างชัดเจน
- มีการระบุจุดเริ่มต้น และ สิ้นสุด เพียงแห่งเดียว

Flow Chart guidelines(2/2)

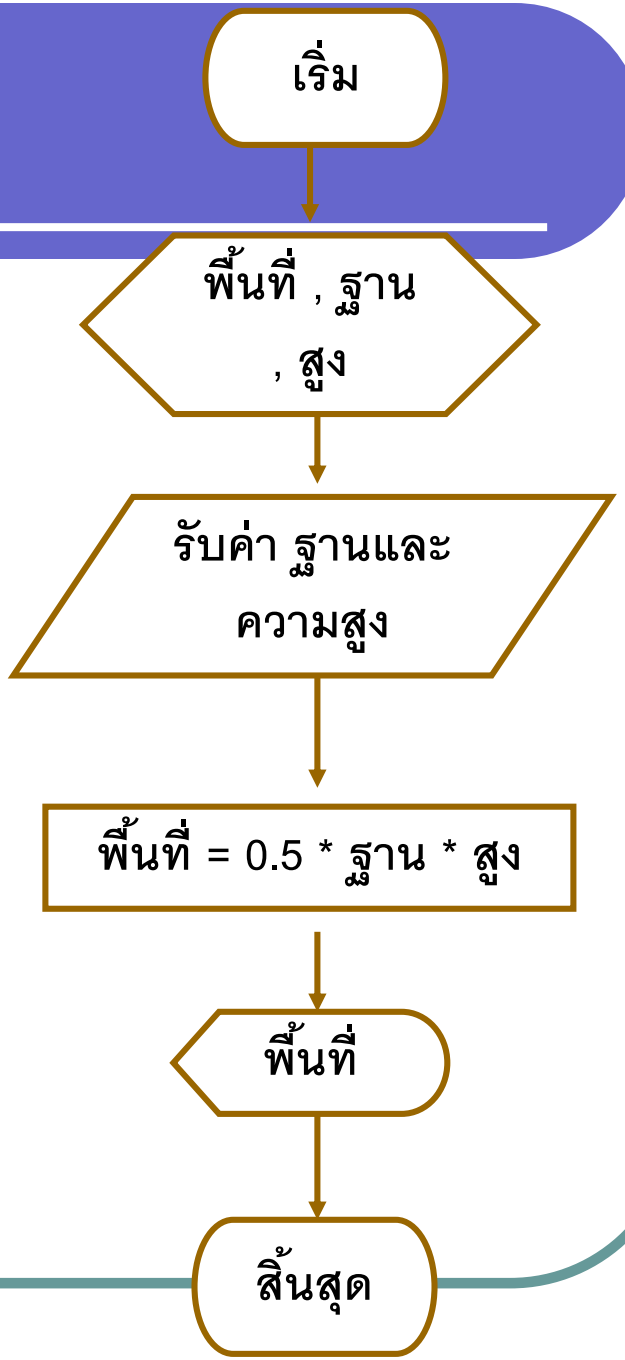
- ใช้คำอธิบายประเภท “one-verb” (verb-noun) เช่น “prepare statement,” “file record”
- กำหนดขั้นตอนไว้ตามลำดับ
- ใช้สัญลักษณ์มาตรฐาน
- เส้นตรงที่ลากระหว่างจุดไม่ควรตัดกัน ในกรณีที่จำเป็นใช้สัญลักษณ์ของการคร่อมสายแทน

ประโยชน์การเขียน Flow Chart(1/2)

- ทำให้มองเห็นภาพการทำงานของตัวโปรแกรมชัดเจนขึ้น
- เป็นตัวกลางในการสื่อสารระหว่างโปรแกรมเมอร์ด้วยกัน
- แปลงความต้องการของผู้ใช้ให้อยู่ในรูปของการทำงานที่เป็นขั้นตอนและง่ายต่อการพัฒนา
- ใช้ในการทำความเข้าใจเกี่ยวกับโปรแกรมเมื่อมีการแก้ไข ปรับปรุงโปรแกรม

- **การหาพื้นที่สามเหลี่ยม**
  - รับค่าความยาวฐานและ ความสูง
  - คำนวณพื้นที่ จากสูตร

    พื้นที่ = 0.5 * ฐาน * สูง
  - แสดงผลลัพธ์ที่ได้จากการคำนวณออกทางจอภาพ



- **คำนวณหาผลรวมของการทำข้อสอบวิชา คพ200 ของนศ. คนหนึ่ง โดยมีคะแนนดังต่อไปนี้**
  - ข้อสอบตัวเลือก 45
  - ข้อสอบอัตนัยข้อ 1 ได้ 4
  - ข้อสอบอัตนัยข้อ 2 ได้ 3
  - ข้อสอบอัตนัยข้อ 3 ได้ 10
  - ข้อสอบอัตนัยข้อ 4 ได้ 4
  - ข้อสอบอัตนัยข้อ 5 ได้ 5

รหัสเทียม(Pseudo code)

เป็นเครื่องมือที่ใช้ในการออกแบบและแสดงอัลกอริทึม ประกอบด้วยคำสั่งที่ไม่ใช่คำสั่งในภาษาโปรแกรมคอมพิวเตอร์ใด ๆ

คุณสมบัติของรหัสเทียม(Pseudo code)

- ง่ายในการเขียน
- ง่ายสำหรับผู้อ่าน
- ไม่มีกฎเกณฑ์ทางไวยากรณ์
- มีโครงสร้างของประโยคคำสั่งที่ดี
- เข้าสู่โปรแกรมคอมพิวเตอร์ภาษาไหนก็ได้ เช่น C, C++ หรือ Java

คำสั่งในรหัสเทียม

- START
- Stop
- END OF THE ALGORITHM
- If *condition* Then *operations* Else *operations*
- If *condition* Then *operations*
- Read *values*
- Write *values*

คำสั่งในรหัสเทียม

- Set the *value* of x to *expression* หรือ x ← *expression*
- Add x to y หรือ y ← y + x
- Subtract x from y หรือ y ← y - x
- Multiply x by y หรือ y ← y * x
- Divide x into y หรือ y ← y / x
- Increment x หรือ x ← x + 1
- Decrement x หรือ x ← x - 1

คำสั่งในรหัสเทียม

- While *condition* do

  :

  :

  End of the loop

- Repeat the following *count* times

  :

  :

  End of the loop

ตัวอย่าง

START

Read X, Y

If X < Y Then

X ↔ Y

EndIf

R ← X mod Y

While R > 0 do

X ← Y

Y ← R

R ← X mod Y

End of the loop

Write “GCD = “, Y

Stop

END OF ALGORITHM